PEMBUATAN TANDA TANGAN DIGITAL MENGGUNAKAN *DIGITAL SIGNATURE ALGORITHM*

Faizah Nurhasanah[1], Raden Sulaiman[1]
[1]Jurusan Matematika, MIPA, Universitas Negeri Surabaya
60231
[1]Jurusan Matematika, MIPA, Universitas Negeri Surabaya
60231
email :fayza_91@yahoo.co.id[1], sulaimanraden@yahoo.com[1]

ABSTRAK

Tanda tangan digital dapat digunakan untuk melakukan pembuktian secara matematis bahwa data tidak mengalami modifikasi secara ilegal, sehingga bisa digunakan sebagai salah satu solusi untuk melakukan verifikasi data. Tujuan dari penulisan makalah ini adalah mengetahui proses pembuatan tanda tangan digital menggunakan *digital signature algorithm.* Proses pembuatan tanda tangan digital diawali dengan pembuatan kunci publik dan kunci privat. Proses pembuatan kunci menghasilkan kunci publik (p,q,g,y) dan kunci privat x. Kunci publik akan dikirimkan kepada penerima pesan untuk memverifikasi tanda tangan. Pada proses perhitungan nilai hash akan dihasilkan *message diggest,* yang akan digunakan dalam pembuatan tanda tangan. Proses penandatanganan dihasilkan sepasang tanda tangan (r,s). Tanda tangan dan dokumen dikirimkan kepada penerima. Selanjutnya, pada proses verifikasi, penerima akan mengecek apakah tanda tangan tersebut cocok atau tidak dengan menggunakan kunci publik dan menghitung nilai hash dokumen yang ia terima.

Keywords: tanda tangan digital, *digital signature algorithm,* fungsi *hash,* kriptografi, tanda tangan digital dengan *digital signature algorithm*

PENDAHULUAN

Perkembangan ilmu dan teknologi dewasa ini telah memengaruhi segala aspek kehidupan. Salah satunya di bidang teknologi informasi pengiriman pesan. Pengiriman pesan menjadi lebih cepat namun perlu diwaspadai tingkat keamanannya karena bisa saja pada saat proses pengiriman pesan ada pihak ketiga yang ingin mengubah pesan tersebut. Salah satu cara untuk mempertahankan kerahasiaan pesan tersebut dengan mengirimkan pesan menjadi kode-kode yang tidak dipahami, sehingga bila ada pihak ketiga yang ingin mengubah akan kesulitan dalam menerjemahkan isi pesan yang sebenarnya. Namun, hanya dengan menyandikan pesan tersebut, tidak menutup kemungkinan pesan dapat diubah oleh pihak ke tiga. Untuk memperkuat kerahasiaan serta keaslian dari pesan tersebut, maka digunakan tanda tangan digital. Penerima pesan akan percaya bahwa pesan yang dikirimkan masih asli, karena telah dibubuhkan tanda tangan digital pada pesan tersebut.

Untuk membuat tanda tangan digital dipergunakan suatu ilmu yang mempelajari teknik – teknik matematika yang berhubungan dengan aspek keamanan informasi, integritas suatu data, serta otentikasi data yaitu kriptografi. Salah satunya kita dapat menggunakan algoritma tanda tangan DSA (*Digital Signature Algorithm*). DSA merupakan salah satu dari algoritma kunci. DSA adalah salah satu algoritma yang digunakan dalam DSS (*Digital Signature Standard*), standard untuk *digital signature* yang dibuat oleh FIPS (*Federal Information Processing Standard*). Pada makalah ini akan membahas tentang proses pembuatan tanda tangan digital dengan DSA.

Kriptografi

Kriptografi (*cryptography*) berasal dari Bahasa Yunani, yaitu *cryptos* berarti rahasia, *graphien* berarti tulisan. Jadi secara asal bahasa, kriptografi berarti *secret writing* (tulisan rahasia). Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, serta otentikasi.

Aspek-aspek keamanan kriptografi adalah sebagai berikut:

a) Kerahasiaan (*confidentiality*), yaitu untuk menjaga isi dari informasi dari pihak-pihak yang tak berhak untuk mendapatkannya.
b) Integritas data (*data integrity*), yaitu menjamin bahwa pesan masih asli dan belum dimanipulasi oleh pihak – pihakyang tidak berhak. Realisasi layanan ini di dalam kriptografi adalah dengan menggunakan tanda tangan digital.
c) Otentikasi (*authentication*), yaitu berhubungan dengan identifikasi. Misalnya, mengidentifikasi suatu kebenaran pihak–pihakyang berkomunikasi (antara pengirim dan penerima pesan)maupunmengidentifikasi kebenaran sumber pesan (pesan berasal dari pengirim yang benar). Layanan ini diwujudkan dengan menggunakan tanda tangan digital.
d) Nirpenyangkalan (*non-repudiation*), yaitu untuk mencegah pihak yang saling berkomunikasi melakukan penyangkalan. Misalkan salah satu dari pihak menyangkal telah mengirim maupun menerima pesan.

Tanda Tangan Digital

Tanda digital adalah suatu mekanisme otentikasi yang mengijinkan pemilik pesan membubuhkan sebuah sandi pada pesannya yang bertindak sebagai tanda tangan. Jadi tanda tangan di sini bukanlah tanda tangan yang di-digitalisasi menggunakan alat scanner namun suatu nilai kriptografis yang bergantung pada pesan dan pengirim pesan.

Skema tanda tangan adalah lima-tuple $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, V)$ yang memenuhi syarat-syarat berikut:

1. $\mathcal{P}$adalah himpunan berhingga pesan yang mungkin.
2. $\mathcal{A}$adalah himpunan berhingga tanda tangan yang mungkin.
3. $\mathcal{K}$ (ruang kunci) adalah himpunan berhingga kunci yang mungkin.
4. Untuk setiap $K \in \mathcal{K}$, terdapat sebuah algoritma penandatanganan $sig_K \in \mathcal{S}$ dan algoritma verifikasi yang bersesuaian $ver_K \in \mathcal{V}$. Setiap $sig_K: \mathcal{P} \to \mathcal{A}$ dan $ver_K: \mathcal{P} \times \mathcal{A} \to \{benar, salah\}$ adalah fungsi-fungsi sedemikian hingga persamaan berikut dipenuhi untuk setiap pesan $x \in \mathcal{P}$ dan untuk setiap tanda tangan $y \in \mathcal{A}$:

$$ver_K(x, y) = \begin{cases} benar & jika\ y = sig(x) \\ salah & jika\ y \neq sig(x) \end{cases}$$

(2000: Douglas)

Fungsi Hash SHA-1

Dalam kriptografi terdapat sebuah fungsi yang digunakan untuk aplikasi keamanan seperti otentikasi dan integritas pesan. Fungsi tersebut adalah fungsi hash. Fungsi hash merupakan suatu fungsi yang menerima barisan hingga dengan panjang sebarang dan mengkonversinya menjadi barisan hingga keluaran yang panjangnya tertentu. Fungsi hash dapat menerima masukan barisan hingga apa saja. Jika barisan hingga menyatakan pesan M, maka sebarang pesan M berukuran bebas dimampatkan oleh fungsi hash H
melalui persamaan : h = H(M).

Keluaran fungsi hash disebut juga nilai hash (*hash-value*) atau ringkasan pesan (*message digest*/MD). Pada persamaan di atas, h adalah nilai hash dari fungsi H untuk masukan M. Dengan kata lain, fungsi hash mengkompresi sebarang pesan yang berukuran berapa saja menjadi nilai hash yang panjangnya selalu tetap. Fungsi hash merupakan fungsi yang bersifat satu arah (*one-way function*) dimana jika dimasukkan data, maka keluarannya berupa sebuah *fingerprint (*sidik jari*)*, *Message Authentication Code (MAC)*. Fungsi ini biasanya diperlukan bila kita menginginkan pengambilan sidik jari dari suatu pesan. Fungsi hash satu arah adalah fungsi yang bekerja satu arah yaitu pesan yang sudah diubah menjadi ringkasan pesan maka tidak dapat dikembalikan lagi menjadi pesan semula.

Fungsi hash H satu arah mempunyai sifat sebagai berikut :

1. Jika diberikan M (M adalah suatu data dalam hal ini berupa pesan), maka H(M) = h mudah dihitung.
2. Untuk setiap h yang dihasilkan, tidak mungkin dikembalikan nilai M sedemikian sehingga H(M)=h. Itulah sebabnya fungsi H dikatakan fungsi Hash satu arah.
3. Jika diberikan M, tidak mungkin mendapatkan M* sedemikian sehingga H(M) = H(M*). Bila diperoleh pesan M* semacam ini maka disebut tabrakan (*collision*).
   Maka sangat sulit menemukan dua pesan yang berbeda yang menghasilkan nilai hash yang sama.
4. Tidak mungkin mendapatkan dua pesan M dan M* sedemikian sehingga H(M) = H(M*).

Sebuah fungsi hash satu arah H(M) beroperasi pada prapeta pesan M dengan panjang sebarang dan mengembalikan nilai hash h yang memiliki panjang tetap. Fungi hash dikembangkan berdasarkan ide sebuah fungsi kompresi. Fungsi satu arah ini menghasilkan nilai hash

berukuran n pada input sebesar b. Input tersebut berupa suatu fungsi kompresi blok pesan dan hasil blok pesan sebelumnya. Sehingga nilai hash suatu blok pesan M adalah : $h_i=f(M_i, h_{i-1})$

dimana : h_i h_i = nilai hash saat ini

M_i = blok pesan saat ini

h_{i-1} =nilai hash blok pesan sebelumnya

SHA (*Secure Hashing Algorithm*) adalah salah satu jenis dari algoritma fungsi hash. SHA terdiri dari 4 macam yaitu SHA-1, SHA-256, SHA-384, SHA-521. Pada skripsi ini digunakan fungsi hash SHA-1 karena mempunyai nilai hash yang paling kecil.

Tabel 1. Macam – macam SHA

Algoritma	**Naskah (bit)**	**Blok (bit)**	**Kata (bit)**	**Nilai hash/*message digest* (bit)**
SHA-1	$< 2^{64}$	**512**	**32**	**160**
SHA-256	$< 2^{64}$	**512**	**32**	**256**
SHA-384	$< 2^{128}$	**1024**	**64**	**384**
SHA-521	$< 2^{128}$	**1024**	**64**	**512**

SHA–1 adalah algoritma hash yang paling banyak digunakan publik. SHA-1 merupakan salah satu jenis dari fungsi hash satu arah. SHA-1 menerima masukan berupa pesan dengan ukuran maksimum 2^{64} bit (2.147.483.648 gigabyte) dan menghasilkan nilai hash dengan panjang 160 bit. Nilai hash kemudian digunakan dalam DSA untuk menghitung tanda tangan digital pesan tersebut. Nilai hash pesan yang sama dapat diperoleh oleh penerima pesan ketika menerima pesan dari pengirim dengan cara memasukkan pesan tersebut pada fungsi SHA-1. SHA-1 dikatakan aman karena secara matematis tidak mungkin menemukan dua pesan yang berbeda yang menghasilkan nilai hash yang sama atau tidak mungkin menemukan pesan aslinya jika diberikan suatu nilai hash-nya.

Pesan M dengan panjang L bit dimana $1 \leq L \leq 2^{64}$. Algoritma pada SHA-1 menggunakan :

1. Pesan yang tersusun pada kata 32-bit sebanyak 80 buah.
2. Lima variabel kerja masing – masing 32-bit.
3. Lima buah nilai hash masing – masing terdiri dari kata 32-bit $W_0, W_1, \dots, W_{79}$. Lima variabel kerja diberi label a, b, c, d, dan e. Kata dari nilai hash diberi label $H_0^{(i)}, H_1^{(i)}, \dots, H_4^{(i)}$ yang akan menampung nilai hash awal $H^{(0)}$ untuk diganti dengan nilai hash yang berurutan setelah blok pesan diproses $H^{(N)}$.

(FIPS PUB 183-3,2008)

***Preprocessing*/ Pemrosesan Awal SHA – 1**

Proses awal SHA–1 yaitu:

1. *Padding* (Penambahan Pesan)

   Sebelum pesan diproses, terlebih dahulu dilakukan penambahan bit atau *padding*. Panjang blok dalam SHA-1 adalah 512 bit, artinya pada setiap prosesnya digunakan panjang pesan berupa barisan hingga 512 bit. Oleh karena itu sebuah pesan harus ditambah agar pesan yang nantinya akan diproses memiliki panjang kelipatan 512. *Padding* dilakukan dengan cara menambahkan nilai bit "1" pada akhir pesan, kemudian ditambahkan k bit "0", dimana k memenuhi:

   $k + [\text{panjang pesan}] + 1 \equiv 448 \bmod 512$

   Tambahkan blok 64 bit yang menyatakan representasi biner dari panjang pesan. Misalkan sebuah pesan "abc" mempunyai panjang 3x8=24 bit. Sehingga pesan ditambah dengan bit "1" dan bit "0" sebanyak 448 - (24+1) = 423. Dapat diilustrasikan pada gambar berikut ini.

$$\underbrace{01100001}_{a}\ \underbrace{01100010}_{b}\ \underbrace{01100011}_{c}\ 1\ \overbrace{00\ldots00}^{423}\ \overbrace{00\ldots011000}^{64}\ \underbrace{}_{l=24}$$

Gambar 1 Pengelompokkan Pesan M

2. *Parsing* (Penguraian pesan)

   Setelah dilakukan *padding* pada pesan M, kemudian uraikan pesan M dalam N blok dengan setiap blok terdiri dari 512 bit, $M^{(1)}, M^{(2)}, \dots, M^{(N)}$. Setiap 512 bit dari blok input terdiri dari 32-bit kata. Setiap kata terdiri dari 32 bit sehingga terdapat 16 kata 32-bit. 32 bit pertama dari blok pesan i dinotasikan $M_0^{(i)}$, 32-bit berikutnya $M_1^{(i)}$, dan seterusnya sampai $M_{15}^{(i)}$.

3. Penetapan Nilai Awal

   Sebelum proses dimulai terlebih dahulu ditetapkan nilai awal atau dapat disebut *kata buffer*. Pada SHA-1 terdapat 5 nilai *buffer*. Nilai awal akan diproses dengan pesan.

Penetapan nilai hash awal $H_0^{(0)}$ yaitu 32-bit *kata* sebanyak lima buah, dalam heksadesimal sebagai berikut :

$H_0^{(0)}$ = 67452301
$H_1^{(0)}$ = efcdab89
$H_2^{(0)}$ = 98badcfe
$H_3^{(0)}$ = 10325476
$H_4^{(0)}$ = c3d2e1f0
(FIPS PUB 183-3, 2008).

Komputasi Hash SHA-1

Komputasi hash SHA-1 menggunakan fungsi logika $f_0, f_1, \ldots, f_{79}$. Masing – masing fungsi f_t dimana $0 \leq t \leq 79$ beroperasi pada 32-bit word, yaitu x, y, z dan menghasilkan keluaran 32-bit. Fungsi $f_t(x,y,z)$ didefinisikan pada persamaan 3.1 yaitu :

$$f_t(x,y,z) = \begin{cases} Ch(x,y,z) = (x \wedge y) \oplus (\sim x \wedge z) & 0 \leq t \leq 19 \\ Parity(x,y,z) = x \oplus y \oplus z & 20 \leq t \leq 39 \\ Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) & 40 \leq t \leq 59 \\ Parity(x,y,z) = x \oplus y \oplus z & 60 \leq t \leq 79 \end{cases} \quad (3.1)$$

SHA-1 menggunakan delapan konstanta kata 32-bit, K_t pada persamaan 3.2 yaitu:

$$K_t = \begin{cases} 5a827999, & 0 \leq t \leq 19 \\ 6ed9eba1, & 20 \leq t \leq 39 \\ 8f1bbcdc, & 40 \leq t \leq 59 \\ ca62c1d6, & 60 \leq t \leq 79 \end{cases} \quad (3.2)$$

Setelah *preprocessing* lengkap, blok pesan $M^{(1)}, M^{(2)}, \ldots, M^{(N)}$ kemudian diproses dengan langkah sebagai berikut :

Untuk i = 1 sampai N lakukan :

1. Persiapkan urutan pesan $\{W_t\}$ yaitu :

$$W_t = \begin{cases} M_t^{(i)}, & 0 \leq t \leq 15 \\ ROTL^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}), & 16 \leq t \leq 79 \end{cases}$$

Dengan ROTL^n (x) adalah operasi *rotate left*, dimana x adalah sebuah kata dan $n \in Z, n < 32$ yang didefinisikan sebagai:

$ROTL^n(x) = (x \ll n) \vee (x \gg (32-n))$

didapatkan dengan membuang n-bit paling kiri dari x dan menggeser hasilnya dengan "0" pada bagian kanan (hasilnya tetap 32-bit).

2. Inisialisasi lima variabel kerja a, b, c, d dan e dengan nilai hash ke (i-1) yaitu:
   a = $H_0^{(i-1)}$
   b = $H_1^{(i-1)}$
   c = $H_2^{(i-1)}$
   d = $H_3^{(i-1)}$
   e = $H_4^{(i-1)}$
3. Untuk t = 0 sampai 79 :
   $T = ROTL^5(a) + f_t(b,c,d) + e + K_t + W_t$
   e = d
   d = c
   c = $ROTL^5(b)$
   b = a
   a = T
4. Hitung nilai hash antara ke – i, $H^{(i)}$ yaitu :

$$H_0^{(i)} = a + H_0^{(i-1)}$$
$$H_1^{(i)} = b + H_1^{(i-1)}$$
$$H_2^{(i)} = c + H_2^{(i-1)}$$
$$H_3^{(i)} = d + H_3^{(i-1)}$$
$$H_4^{(i)} = e + H_4^{(i-1)}$$

Setelah empat langkah di atas diulang sebanyak N kali akan menghasilkan 160 bit *message digest* dari pesan M yaitu:

$$H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)}$$

HASIL DAN PEMBAHASAN

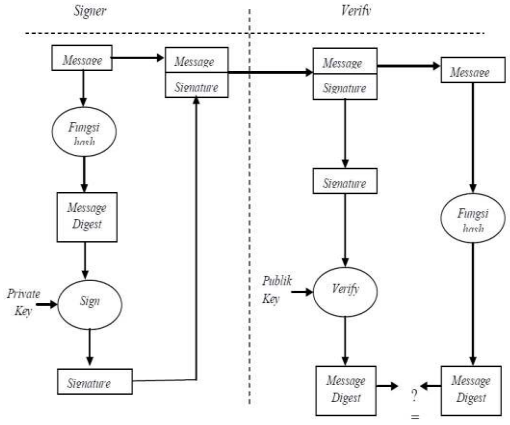
Pada beberapa kasus seringkali otentikasi yang diperlukan tetapi kerahasiaan pesan tidak. Maksudnya, pesan tidak perlu dienkripsikan, sebab yang dibutuhkan hanya keotentikan pesan saja. Kebutuhan tersebut dapat dipenuhi dengan pemberian tanda tangan digital. Algoritma kunci publik dan fungsi hash dapat digunakan untuk kasus seperti ini. Tandatangan digital adalah suatu nilai kriptografis yang bergantung pada pesan dan pengirim pesan. Pada Agustus 1991, NIST (*The National of Standard and Technology*) mengumumkan standard untuk tandatangan *digital* yang dinamakan *Digital Signature*

Standard (DSS) yang terdiri dari dua komponen :

a. Algoritma tandatangan digital yang disebut DSA (*Digital Signature Algorithm)*

b. Fungsi Hash yang disebut SHA (*Secure Hash Algorithm*).

Jadi DSA untuk penandatanganan pesan dan SHA untuk membangkitkan *message digest* dari pesan. Pada makalah ini menggunakan SHA-1.



Gambar 2 Diagram Proses Tanda Tangan Digital

Menentukan Parameter DSA

a. p adalah bilangan prima dengan panjang L bit, dimana $2^{L-1} < p < 2^L$ dengan $512 \leq L \leq 1024$ dan L adalah kelipatan 64. Parameter p bersifat publik dan dapat digunakan bersama-sama oleh orang di dalam kelompok.

b. q adalah bilangan prima 160 bit, faktor dari p-1 dengan kata lain (p-1) mod q = 0. Parameter q bersifat publik dan dapat digunakan bersama-sama oleh orang di dalam kelompok dimana $2^{159} < q < 2^{160}$.

c. Pembangkit g pada grup siklik tunggal dengan order q di dalam Z_p^*. Hitung $g = h^{(p-1)/q} \bmod p$, dengan $h \in Z_p^*$, $1 < h < p-1$ sehingga $g > 1$. Parameter g bersifat publik dan dapat digunakan bersama-sama oleh orang di dalam kelompok.

d. x bilangan bulat dimana $0 < x < q$ dengan panjang 160 bit. Parameter x bersifat privat yang hanya boleh diketahui oleh pengirim pesan.

e. $y = g^x \bmod p$ adalah kunci publik.

f. M adalah pesan yang akan diberi tandatangan.

g. k adalah bilangan bulat yang dibangkitkan random atau pseudorandom dimana $0<k< q$. Parameter k bersifat privat yang hanya boleh diketahui oleh pengirim pesan.

Parameter p, q dan g bersifat publik dan dapat digunakan bersama dalam sekelompok orang. Parameter p, q dan g juga bernilai tetap untuk periode/waktu tertentu. Parameter x dan k hanya digunakan untuk pembangkitan tandatangan dan harus dijaga kerahasiaannya. Parameter k harus berbeda untuk setiap tandatangan.

Pembentukan Sepasang Kunci

Input : Bilangan prima p dan q, dimana (p-1) mod q = 0, elemen primitif g, dimana $g > 1$ dan bilangan $x < q$.

Output : Kunci publik (p,q,g,y) dan kunci privat (p,q,g,x)

Langkah :

1. Pilih bilangan prima p dan q, dimana (p-1) mod q = 0.
2. Hitung $g = h^{(p-1)/q} \bmod p$, dimana $1 < h < p-1$ dan $g > 1$.
3. Tentukan kunci privat $x < q$.
4. Hitung kunci publik $y = g^x \bmod p$.
5. Publikasikan p,q,g, dan y, tetapi nilai x dirahasiakan.

Kunci publik yang dihasilkan pada pembentukan kunci ini bersifat tunggal, karena ketiga nilai yang akan digunakan pada pembentukan kunci sudah ditetapkan, sehingga nilai y adalah tunggal.

Terdapat y_1 dan y_2. Diambil bilangan prima p dan q, dimana (p – 1) mod q = 0,

$g = h^{(p-1)/q} \bmod p$, dimana $1 < h < p-1$ dan $g > 1$ dan $x < q$.

Dengan $y = g^x \bmod p$, maka

$y_1 = g^x mod\ p$ (*) dan $y_2 = g^x mod\ p$ (**)

$y_1 = g^x mod\ p$, maka $g^x = y_1 mod\ p$ (***)

(***) disubstitusikan ke dalam persamaan (**), sehingga

$y_2 = (y_1 mod\ p)\ mod\ p$

$y_2 = y_1 mod\ p$ (berdasarkan aturan aritmatika modular)

Dengan demikian $y_2 = y_1$, terbukti bahwa kunci y bersifat tunggal.

Proses Penandatanganan

Input : Pesan M yang akan dikirimkan, dan kunci privat x.
Output : Pesan M dan tanda tangan (r,s)
Langkah :

1. Ubah pesan M menjadi *nilai hash* dengan fungsi Hash SHA-1menghasilkan SHA-1(M).
2. Tentukan bilangan acak k < q.
3. Tanda tangan dari pesan m adalah bilangan r dan s yang didapat dari :
   $r = (g^k \bmod p) \bmod q$, $s = (k^{-1} (\text{SHA-1}(M) + xr)) \bmod q$
   k^{-1} adalah invers dari k modulo q, dengan $k.k^{-1} \equiv 1 \; mod \; q$.
   Pada perhitungan nilai s, 160-bit barisan hingga SHA-1(M) dikonversi terlebih dahulu ke dalam bilangan bulat. Jika tandatangan yang dihasilkan benar maka nilai r dan atau s tidak mungkin 0.
4. Kirim pesan beserta tandatangan r dan s.

Proses Verifikasi

Setelah pesan telah sampai kepada pihak penerima, maka penerima akan melakukan proses verifikasi. Untuk melakukan proses ini, penerima pesan menggunakan kunci publik (p,q,g,y) yang telah diberikan dari pengirim pesan. Penerima memperoleh pesan yang berupa dokumen yang telah dibubuhi tanda tangan digital. Dalam hal ini, dokumen bisa saja berupa *plaintext* maupun *ciphertext.* Tergantung dari perjanjian antar pihak yang bersangkutan. Sebelumnya akan dihitung terlebih dahulu nilai *hash* dari dokumen yang diterima. Kemudian penerima akan memverifikasi tanda tangan (r, s). terlebih dahulu ia akan mengecek $0 < r < q$ and $0 < s < q$ kemudian menghitung :

$w = s^{-1} \bmod q$
$u_1 = (\text{SHA-1}(M) * w) \bmod q$
$u_2 = (r * w) \bmod q$
$v = ((g^{u1} * y^{u2}) \bmod p) \bmod q)$

Jika v = r maka tandatangan sah, pesan yang diterima dikirim oleh pihak memegang kunci rahasia x sesuai dengan y kunci publiknya, dengan kata lain pesan masih asli dan dokumen dikirim oleh pengirim yang benar.

Jika $v \neq r$, maka terdapat beberapa kemungkinan yaitu pesan telah dimodifikasi, pesan telah salah ditandatangani oleh penandatangan, atau pesan telah ditandatangani oleh pihak lain (bukan penandatangan sebenarnya) berarti pesan tidak valid.

Lemma 1 :
Jika p dan q prima dan q pembagi (p-1), h bilangan bulat positif lebih kecil dari p dan
$g = h^{(p-1)/q} \bmod p$ maka $g^q \bmod p = 1$ dan jika m mod q = n mod q maka $g^m \bmod p = g^n \bmod p$.
Bukti :

$$g^q \bmod p = (h^{(p-1)/q} \bmod p)^q \bmod p$$
$$= h^{(p-1)} \bmod p$$
$$= 1$$

dengan Teorema Little Fermat (*Fermat's Little Theorem*).
Jika m mod q = n mod q, maka , m = n + kq untuk beberapa bilangan bulat k sehingga :

$$g^m \bmod p = g^{n+kq} \bmod p$$
$$= (g^n g^{kq}) \bmod p$$
$$= ((g^n \bmod p) (g^q \bmod p)^k) \bmod p$$
$$= g^n \bmod p$$

Jadi terbukti $g^q \bmod p = 1$.

Teorema 1 : (Pembuktian v = r')
Jika M' = M, r' = r, dan s' = s pada verifikasi tandatangan, maka v = r'.
Bukti :
$w = (s')^{-1} \bmod q = s^{-1} \bmod q$
u1 = ((SHA-1(M'))w) mod q = ((SHA-1(M))w) mod q
u2 = ((r')w) mod q = (rw) mod q.
Diketahui $y = g^x \bmod p$, dengan lemma 3.1 diperoleh :
$v = ((g^{u1} y^{u2}) \bmod p) \bmod q$
$= ((g^{\text{SHA-1}(M)w} y^{rw}) \bmod p) \bmod q$, $y = g^x \bmod p$
$= ((g^{\text{SHA-1}(M)w} g^{xrw}) \bmod p) \bmod q$
$= ((g^{(\text{SHA-1}(M)+xr)w}) \bmod p) \bmod q$.
Juga
$s = (k^{-1}(\text{SHA-1}(M) + xr)) \bmod q$.
Sehingga
$w = (k(\text{SHA-1}(M) + xr)^{-1}) \bmod q$
$(\text{SHA-1}(M) + xr)w \bmod q = k \bmod q$.
Dengan lemma 3.1 didapat :
$v = (g^k \bmod p) \bmod q$
= r = r'.

(FIPS PUB 186-3, 2009)

SIMPULAN

Kesimpulan yang diperoleh dari makalah ini yaitu bahwa tanda tangan digital DSA dengan SHA-1 merupakan salah satu alternatif dalam pengamanan data sehingga pihak-pihak yang berkaitan terhadap data tersebut dapat merasa yakin bahwa data tersebut aman. Tingkat keamanan dari tanda tangan ini terletak pada kesulitan proses SHA-1 dan tingkat kesulitan pencarian kunci privat sehingga kecil kemungkinan adanya pencurian data dari pihak-pihak yang tidak diinginkan.

SARAN

Diharapkan pada penelitian selanjutnya untuk membahas tentang cara menyelesaikan permasalahan logaritma diskrit secara mendalam dan juga dapat membahas aplikasi dari tanda tangan DSA.

DAFTAR PUSTAKA

[1] Arahman, Mahdi. 2006. *Pembuatan Sistem Tanda Tangan Digital dengan Menggunakan Algoritma Kurva Elliptic dan Fungsi Hash SHA-512*. Skripsi, Matematika, ITS, Surabaya.

[2] Arizka, Rininda Ulfa. 2011. *Penerapan Sistem Kriptografi Elgamal Atas Z_p* Dalam Pembuatan Tanda Tangan Digital*. Skripsi, Matematika, UNY, Yogyakarta.

[3] Buchmann, Johannes A. 2000. *Introduction to Cryptography*. New York : Springer-Verlag.

[4] Kromodimoeljo, Sentot. 2009. *Teori dan Aplikasi Kriptografi.* SPK IT Consulting.

[5] Maryanto, Budi. 2008. *Penggunaan Fungsi Hash Satu Arah Untuk Enkripsi Data.* Jurnal, Informatika, STMIK LIKMI, Bandung.

[6] Menezes, Alfred J., Ooschot, Paul C. Van, Vanstone, Scott A., 1997. *Handbook of Applied Cryptography*, CRC Pres LLc, Boca Raton,.

[7] Munir, R**.** 2006. *Kriptografi*. Informatika, Bandung.

[8] Priwardani, Restia & Rahayu, Intan. *Perbandingan Skema Tanda Tangan Digital ElGamal, DSA dan Schnorr*.

[9] Sukirman. 2006. *Pengantar Teori Bilangan.* Yogyakarta : Hanggar Kreator.

[10] Stinson R. Douglas. 2000. *Cryptography Theory Practice.* CRC Press: Tokyo.

[11] Wahyuni, Ana. 2011. *Aplikasi Kriptografi untuk Pengamanan e-dokumen dengan Metode Hybrid: Biometrik Tandatangan dan DSA (Digital Signature Algorithm).* Tesis, Sistem Informasi, UNDIP, Semarang.

[12] NIST, FIPS PUB 186-3 *Digital Signature Standard (DSS)*, 2009